REMARKS

Claims 1-71 are pending in the application. All Claims have been rejected.

In response to the Examiner's objection to the drawings, FIG. 1 of the present application was amended to include reference numerals associated with the text of FIG. 1. Amended FIG. 1 is included herewith. Related sections of the specification were similarly amended to include the reference numbers of FIG. 1. No new matter was added.

The Examiner rejected Claims 4, 11, 18, 23, 26-37, 41, 48, and 57 under 35 U.S.C. §112, second paragraph, as being indefinite and Claims 26-37 under 35 U.S.C. §101, as being directed to non-statutory subject matter. All informalities and deficiencies alleged by the Examiner with reference to 35 U.S.C. §112 and §101 have been addressed herein.

The Examiner rejected Claims 1-71 under 35 U.S.C. §102, as being anticipated by U.S. Patent No. 6,085,035 (Ungar).

The present invention is directed to the detection of mutability of fields and classes in an arbitrary program component. The definition of "mutability" of the present invention is fundamentally different from that given by Ungar. The key part of the mutable/immutable definition of Ungar is may be found in column 5, lines 5-9 of Ungar. That definition reads as follows:

> "Some variables store data-values of only one type. These variables are immutable type variables (all statically typed variables are immutable type variables). Mutable type variables store data values of different types."

In its preferred embodiment, Ungar further describes how this definition can be used for compiler / runtime optimizations.

The present invention defines mutability as a change in *any* instance of type after a time t. The invention identifies whether *any* instance of type is immutable (unchanged) after a time t. Typically identification is performed after an instance of a type is initialized via its instance constructor method. More specifically, the present invention identifies whether the state of any of the variables of an instance can be mutated (changed) after time t. This definition holds irrespective of whether the data type of the object referenced by a variable is of the same or different type.

As defined in Ungar, the type remains immutable if a new reference is stored in a variable, however the new and old references must be of the same type. In contrast, the present invention, because of its definition of mutability, provides that the variable is mutable and therefore the type (class) of the variable is mutable as well.

Furthermore, Ungar does not include a notion of a time t after which an object's state may no longer be modified if it is to be considered immutable. This is introduced by the present invention.

The following example contrasts the two definitions of immutability / mutability. In Example 1 below, variable b is immutable according to Ungar (a variable is type-immutable if during its lifetime it stores objects of one type only) because it always references objects of type Boolean. The variable b is *not* immutable according to the definition of the present invention (a variable is immutable if from some defined time t onward its state, i.e., its value and values of all the variables reachable from it are constant). Class B would be considered mutable according to the present invention since its state (variable b) is modified by an instance method.

Conversely, in the following code example, variable a is immutable according to the present invention because the state of an instance of class A is not changed once an instance of A is constructed (time t is the end of the constructor method A()). In contrast, variable a is mutable according to Ungar since it may reference different types of objects (Boolean and Integer).

Example 1
```
public class B {
  Object b;
  public void methodB() {
    b = new Boolean (true);
    b = new Boolean(false);
  }
}
```

Example 2
```
public class A{
  private Object a;
  A(){
    a = new Boolean(true);
    a = new Integer(5);
  }
}
```

In view of the above Ungar does not teach or describe a method of detecting mutability recited in Claims 1, 7, 8, and 19 of the present invention; a device for detecting mutability recited in Claims 26 of the present invention; and a computer system for detecting mutability recited in Claims 38, 44, 45, 53, 60 of the present invention.

Accordingly, it is submitted that independent Claims 1, 7, 8, 19, 26, 38, 44, 45, 53, and 60 are patentable over Ungar. Without conceding the patentability per se of dependent Claims 2-6, 9-25, 27-43, 46-52, 54-59, and 61-71, these are likewise believed to overcome rejection by virtue of their dependence on their respective independent claims.

Applicants believe that all Claims pending in the present application are in condition for allowance. If the Examiner has any questions regarding this communication or feels that an interview would be helpful in advancing the prosecution of this application, the Examiner is requested to contact the undersigned attorney.

Respectfully submitted,

[illegible]

Paul J. Farrell
Reg. No. 33,494
Attorney for Applicant

DILWORTH & BARRESE, LLP
333 Earle Ovington Boulevard
Uniondale, New York 11553
Telephone: 516-228-8484
Facsimile: 516-228-8516